Practical 1
Time Series analysis of Unemployment level

1.Data Introduction

Source: https://fred.stlouisfed.org/series/UNEMPLOY

U.S. Bureau of Labor Statistics (BLS)

Units: Thousands of Persons, Seasonally Adjusted

Frequency: Monthly

The series comes from the 'Current Population Survey (Household Survey)'(CPS)
This data set contains monthly time series data spanning from January 1948 to October 2024. Data contains 2 columns and 932 rows in which one column is for date and the other column is for the Unemployment level (Thousands of Persons)

The Unemployment Level is the aggregate measure of people currently unemployed in the US. Someone in the labor force is defined as unemployed if they were not employed during the survey reference week, were available for work, and made at least one active effort to find a job during the 4-week survey period.

The Unemployment Level is collected in the CPS and published by the BLS. It is provided on a monthly basis, so this data is used in part by macroeconomists as an initial economic indicator of current trends. The Unemployment Level helps government agencies, financial markets, and researchers gauge the overall health of the economy.

The individuals that are not employed but not actively looking for a job are not counted as unemployed. For instance, declines in the Unemployment Level may either reflect movements of unemployed individuals into the labor force because they found a job, or movements of unemployed individuals out of the labor force because they stopped looking to find a job.

2.Upload the data and convert it into time series data

R-Code:

```
install.packages("forecast")
library(forecast)
data=read.csv("C:/Users/91911/Downloads/UNEMPLOY.csv")
data
```

Output:

(As the data is very large this is only a glimpse of the data)

```
      DATE UNEMPLOY
1  1948-01-01    2034
2  1948-02-01    2328
3  1948-03-01    2399
```

4 1948-04-01 2386
5 1948-05-01 2118
6 1948-06-01 2214
7 1948-07-01 2213
8 1948-08-01 2350
9 1948-09-01 2302
10 1948-10-01 2259
11 1948-11-01 2285
12 1948-12-01 2429
13 1949-01-01 2596
14 1949-02-01 2849
15 1949-03-01 3030
16 1949-04-01 3260
17 1949-05-01 3707
18 1949-06-01 3776
19 1949-07-01 4111
20 1949-08-01 4193

R-Code:

```
# Create a time series object
ts_data = ts(data$UNEMPLOY, start = c(1948,1) , frequency = 12);ts_data
```

Output:

(As the data is very large this is only a glimpse of the data)

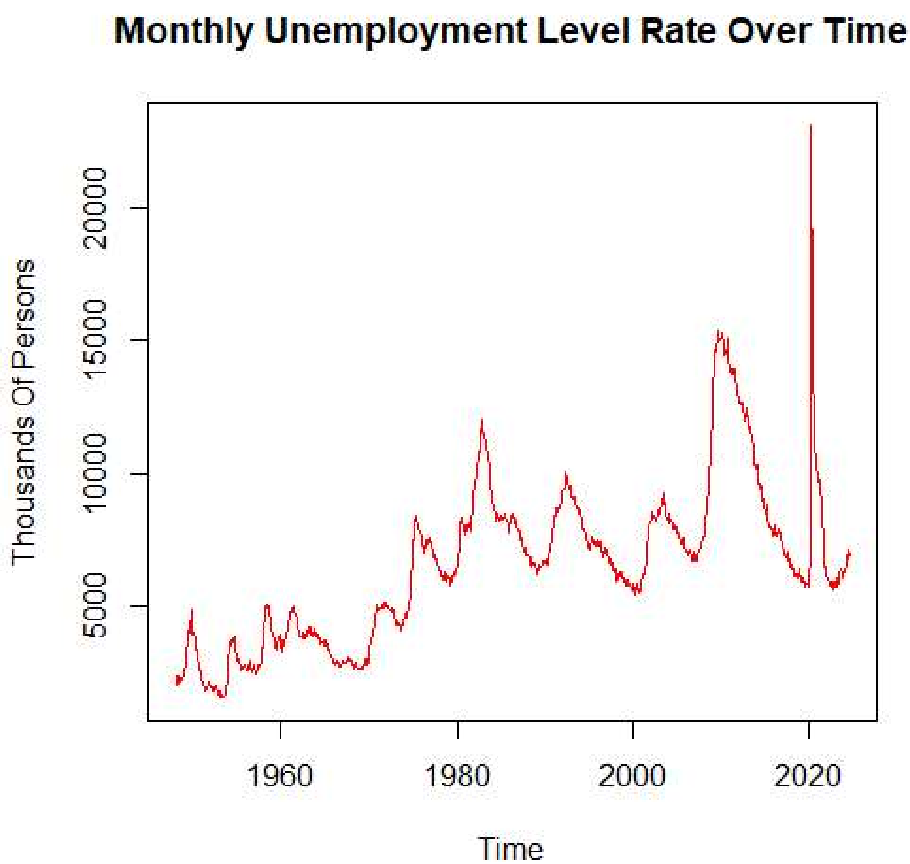
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1948	2034	2328	2399	2386	2118	2214	2213	2350	2302	2259	2285	2429
1949	2596	2849	3030	3260	3707	3776	4111	4193	4049	4916	3996	4063
1950	4026	3936	3876	3575	3434	3367	3120	2799	2774	2625	2589	2639
1951	2305	2117	2125	1919	1856	1995	1950	1933	2067	2194	2178	1960
1952	1972	1957	1813	1811	1863	1884	1991	2087	1936	1839	1743	1667
1953	1839	1636	1647	1723	1596	1607	1660	1665	1821	1974	2211	2818
1954	3077	3331	3607	3749	3767	3551	3659	3854	3927	3666	3402	3196
1955	3157	2969	2918	3049	2747	2701	2632	2784	2678	2830	2780	2761
1956	2666	2606	2764	2650	2861	2882	2952	2701	2635	2571	2861	2790
1957	2796	2622	2509	2600	2710	2856	2796	2747	2943	3020	3454	3476
1958	3875	4303	4492	5016	5021	4944	5079	5025	4821	4570	4188	4191
1959	4068	3965	3801	3571	3479	3429	3528	3588	3775	3910	4003	3653
1960	3615	3329	3726	3620	3569	3766	3836	3946	3884	4252	4330	4617
1961	4671	4832	4853	4893	5003	4885	4928	4682	4676	4573	4295	4177
1962	4081	3871	3921	3906	3863	3844	3819	4013	3961	3803	4024	3907
1963	4074	4238	4072	4055	4217	3977	4051	3878	3957	3987	4151	3975
1964	4029	3932	3950	3918	3764	3814	3608	3655	3712	3726	3551	3651
1965	3572	3730	3510	3595	3432	3387	3301	3254	3216	3143	3073	3031
1966	2988	2820	2887	2828	2950	2872	2876	2900	2798	2798	2770	2912
1967	2968	2915	2889	2895	2929	2992	2944	2945	2958	3143	3066	3018

3.Plot the time series data

R-Code:

```
plot(ts_data, type = "l", col = "red", xlab = "Time", ylab = "Thousands Of Persons", main = "Monthly Unemployment Level Rate Over Time")
```

Output:



Interpretation :
The graph shows a pattern where unemployment rises during tough economic times (recessions) and falls during recoveries.Here we see that the biggest jump in the graph is around 2020, caused by the COVID-19 pandemic, which led to many people losing their jobs and After each spike, the number of unemployed people goes down, showing that the economy eventually recovers and more people find jobs.

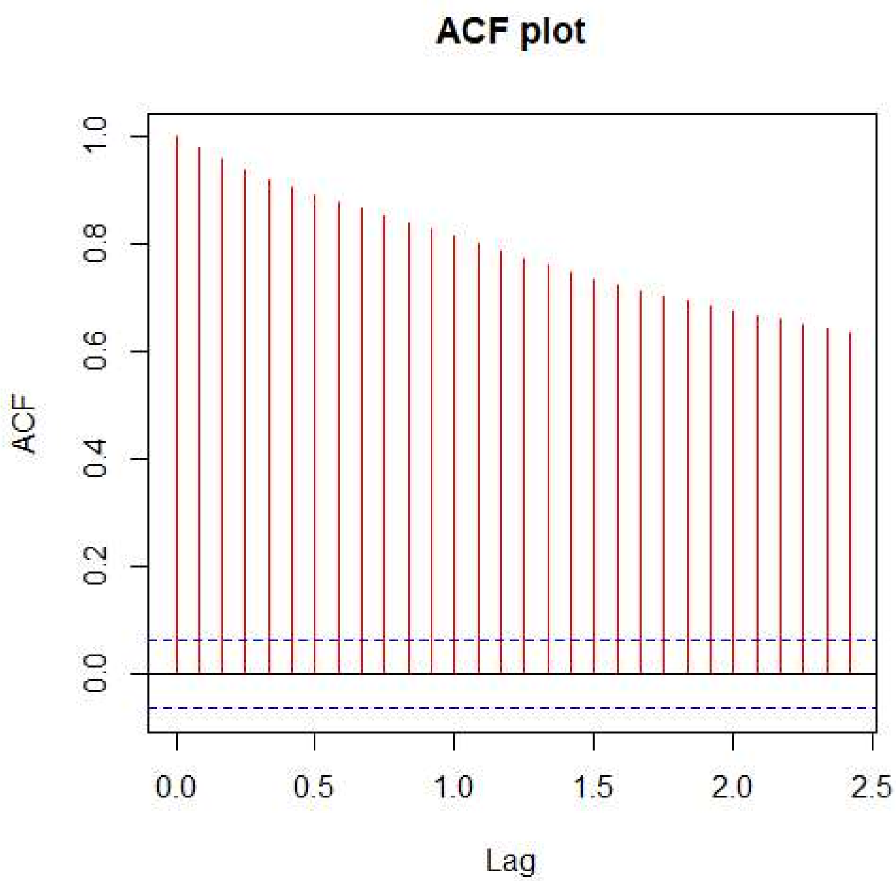
4.Autocorrelation plot

The ACF plot displays the autocorrelation coefficients (values) at different lags. It's a graphical representation that shows how the autocorrelation changes as the lag increases.

R-code:

```
acf(ts_data,main="ACF plot", col = "red")
```

Output:



Interpretation:

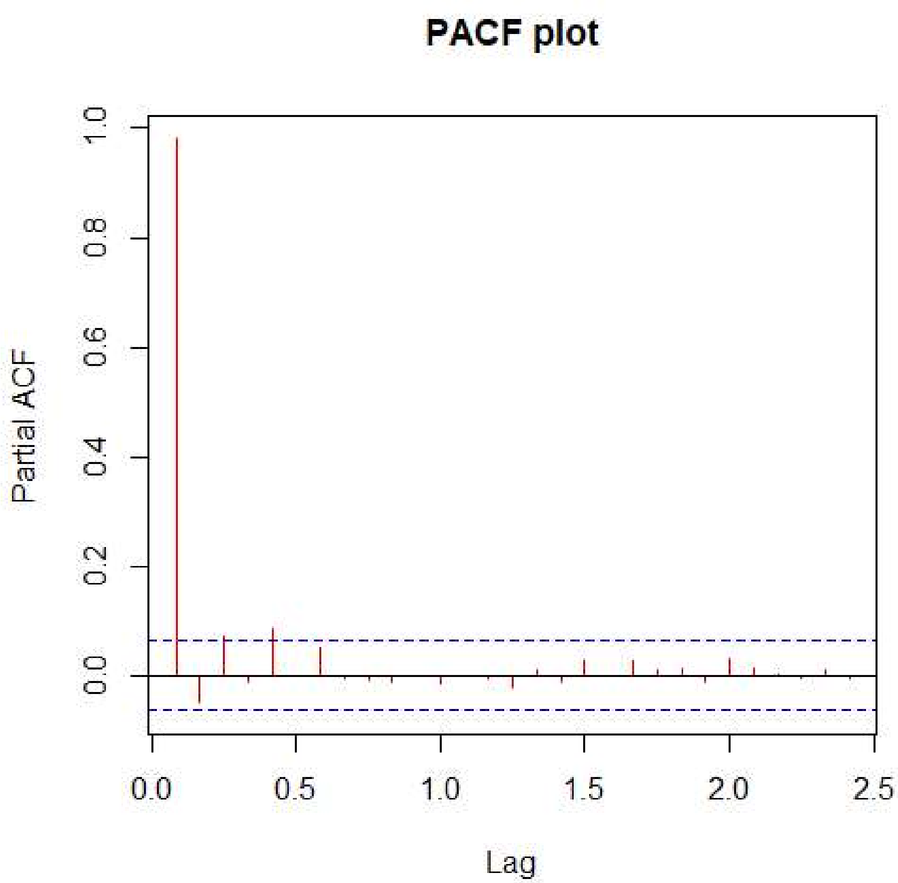
The plot shows that the current unemployment rate is strongly related to the rates in the recent past. As time goes on, this relationship weakens but still exists. This means that if the unemployment rate is high now, it's likely to remain high for the next few months, gradually becoming less predictable as time goes on.

5.Partial autocorrelation function plot

R-code:

```
pacf(ts_data,main="PACF plot",col="red")
```

Output:



Interpretation:
The plot shows that the unemployment rate this month is directly and strongly influenced by the rate last month.However, once we take the unemployment rate of the previous month, the rates from earlier months don't have much additional direct impact.

6.Stationarity test

R-Code:

```
install.packages("tseries")
library(tseries)
adf_result=adf.test(ts_data);adf_result
if(adf_result$p.value<0.05)
{
cat("The time series is stationary(reject the null hypothesis).\n")
}else{
cat("The time series is non stationary(fail to reject the null hypothesis).\n")
```

}

Output:
Augmented Dickey-Fuller Test

data: ts_data
Dickey-Fuller = -3.4292, Lag order = 9, p-value = 0.04911
alternative hypothesis: stationary

The time series is stationary(reject the null hypothesis).

Interpretation:
In order to test the stationarity of a time series, we can use the **Augmented Dickey-Fuller Test** (adf test). A p-Value of less than 0.05 in *adf.test()* indicates that the time series is stationary. Here we see that the p-value is less than 0.05 that means the time series is stationary.

7.Durbin-Watson (DW) test

The Durbin-Watson (DW) test is a statistical test used to detect the presence of autocorrelation in the residuals .The DW test specifically looks for autocorrelation at lag 1, which is the correlation between consecutive residuals.

R-Code:

```
ar1_model = Arima(ts_data, order = c(1, 0, 0), seasonal = c(0, 0, 0))
residuals_ar1 = residuals(ar1_model)
dw_test_result = dwtest(residuals_ar1 ~ 1)
print(dw_test_result)
```

Output:
Durbin-Watson test

data: residuals_ar1 ~ 1
DW = 1.8954, p-value = 0.05594
alternative hypothesis: true autocorrelation is greater than 0

Interpretation:
The DW statistic of 1.8954 indicates that there is little to no autocorrelation in the residuals of your AR(1) model.
The p-value of 0.05594 is marginally above the 0.05 threshold, implying that we do not have enough evidence to conclude there is significant autocorrelation. However, it is close enough to warrant some caution, as there may be weak evidence of positive autocorrelation.

8.Fitting an AR model

R-Code:

```
results=data.frame(Lag=integer(),AIC=numeric(),BIC=numeric(),AR_coefficient=list())
for(lag in 1:5){
ar_model=Arima(ts_data,order=c(lag,0,0))
AR_AIC=AIC(ar_model)
AR_BIC=BIC(ar_model)
AR_coefs=coef(ar_model)
results = rbind(results, data.frame(Lag = lag, AIC = AR_AIC ,BIC = AR_BIC, Coefficients =
I(list(AR_coefs))))
}
results
min_aic_ar_model =results[which.min(results$AIC), ];min_aic_ar_model
min_bic_ar_model = results[which.min(results$BIC), ];min_bic_ar_model
write.csv(results,file='AR_result.csv')
```

Output:

```
Lag      AIC      BIC Coefficients
1   1 14390.46 14404.94 0.981051....
2   2 14389.80 14409.10 1.034011....
3   3 14386.46 14410.60 1.038120....
4   4 14388.34 14417.30 1.038741....
5   5 14382.14 14415.92 1.039777....
> min_aic_ar_model =results[which.min(results$AIC), ];min_aic_ar_model
  Lag      AIC      BIC Coefficients
5   5 14382.14 14415.92 1.039777....
> min_bic_ar_model = results[which.min(results$BIC), ];min_bic_ar_model
  Lag      AIC      BIC Coefficients
1   1 14390.46 14404.94 0.981051....
```

	Lag	AIC	BIC	Coefficients		
1	1	14390.4585	14404.93814	c(ar1 = 0.981051220385521	intercept = 6570.00120975307)	
2	2	14389.79617	14409.10235	c(ar1 = 1.03401151425367	ar2 = -0.0537803691859822	intercept = 6591.84550617093)
3	3	14386.46254	14410.59526	c(ar1 = 1.03812015634067	ar2 = -0.132331207427913	ar3 = 0.0759555482510831

4	4	14388.33923	14417.2985	c(ar1 = 1.03874149967272	ar2 = -0.133477236132087	ar3 = 0.0865613323043966
5	5	14382.13875	14415.92456	c(ar1 = 1.03977764128252	ar2 = -0.141783754519991	ar3 = 0.0991169904131808

intercept = 6592.29507262326)		
ar4 = -0.0102252659452373	intercept = 6516.87229794611)	
ar4 = -0.107752181035061	ar5 = 0.0938993328432927	intercept = 6407.87223721629)

Interpretation:

The results showed multiple autoregressive (AR) models with different lags and compared their Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) values along with the model coefficients as shown in the output.

AIC (Akaike Information Criterion):

- The model with the minimum AIC is considered to have the best fit in terms of balancing model complexity and goodness of fit.
- In this case, the model with lag 5 has the minimum AIC.

BIC (Bayesian Information Criterion):

- The model with the minimum BIC is generally preferred when we want a model that avoids overfitting. BIC imposes a larger penalty for additional parameters than AIC.
- In this case, the model with lag 1 has the minimum BIC.

9.Fitting an MA model

R-Code:

```
results2=data.frame(Lag2=integer(),MA_AIC=numeric(),MA_BIC=numeric(),MA_coefficient=l
ist())
for(lag2 in 1:5){
ma_model=Arima(ts_data,order=c(0,0,lag2))
ma_AIC=AIC(ma_model)
ma_BIC=BIC(ma_model)
ma_coefs=coef(ma_model)
results2 = rbind(results2, data.frame(Lag2 = lag2, MA_AIC = ma_AIC ,MA_BIC = ma_BIC,
Coefficients = I(list(ma_coefs))))
}
results2
min_aic_ma_model =results2[which.min(results2$MA_AIC), ];min_aic_ma_model
min_bic_ma_model = results2[which.min(results2$MA_BIC), ];min_bic_ma_model
write.csv(results2,file='MA_result.csv')
```

Output:

```
 Lag2   MA_AIC   MA_BIC Coefficients
1    1 16338.18 16352.66 0.899063....
2    2 15729.15 15748.45 1.318455....
3    3 15306.17 15330.30 1.413943....
4    4 15090.88 15119.84 1.420849....
5    5 14927.34 14961.13 1.371626....
> min_aic_ma_model =results2[which.min(results2$MA_AIC), ];min_aic_ma_model
  Lag2   MA_AIC   MA_BIC Coefficients
5    5 14927.34 14961.13 1.371626....
> min_bic_ma_model = results2[which.min(results2$MA_BIC), ];min_bic_ma_model
  Lag2   MA_AIC   MA_BIC Coefficients
5    5 14927.34 14961.13 1.371626....
```

	Lag2	MA_AIC	MA_BIC	Coefficients	
1	1	16338.17848	16352.65811	c(ma1 = 0.899063836821988	intercept = 6592.2654055804)
2	2	15729.14596	15748.45214	c(ma1 = 1.31845573912645	ma2 = 0.670072963421679
3	3	15306.17061	15330.30334	c(ma1 = 1.41394361577205	ma2 = 1.18987397665404
4	4	15090.88085	15119.84012	c(ma1 = 1.42084956798776	ma2 = 1.36010057108854

5	5	14927.34311	14961.12893	c(ma1 = 1.37162612971378	ma2 = 1.41803461222669

intercept = 6591.80011977597)			
ma3 = 0.589049541498059	intercept = 6590.48679775368)		
ma3 = 1.02533419558362	ma4 = 0.429417765881427	intercept = 6588.54726192214)	
ma3 = 1.25650370533669	ma4 = 0.832833303365832	ma5 = 0.382332066282928	intercept = 6585.98828451081)

Interpretation:
I have evaluated MA models with lag values ranging from 1 to 5. For each model, I have calculated the AIC and BIC values along with the model coefficients as shown in the output. Here we see that at lag 5 the AIC and BIC value is minimum, so we selected a model with lag 5 because a lower AIC and BIC indicates a better fit relative to other models, considering both the complexity and the goodness of fit.

10.Fitting an ARIMA model

R-code:
model=auto.arima(ts_data);model

Output:
```
Series: ts_data
ARIMA(2,1,2)

Coefficients:
        ar1     ar2      ma1      ma2
     0.2611  0.6572  -0.2093  -0.7668
s.e.  0.0985  0.0965   0.0848   0.0842
```

```
sigma^2 = 343978:  log likelihood = -7175.66
AIC=14361.32   AICc=14361.39   BIC=14385.45
>
```

Interpretation:

ARIMA(2,1,2): This indicates the model includes:

a. **2 autoregressive (AR) terms (ar1 and ar2)**
   (ar1= 0.2611,standard error = 0.0985)
   (ar2=0.6572 ,standard error = 0.0965)

b. **1 differencing term (indicated by d=1).**

c. **2 moving average (MA) terms (ma1 and ma2)**
   (ma1=-0.2093 ,standard error =0.0848
   (ma2=-0.7668,standard error =0.0842)

   An ARIMA(2,1,2) model has been fitted to the data, meaning the model uses two past values of the series and two past forecast errors to predict the current value, after differencing the series once to make it stationary.

   sigma^2 = 343978: This is the estimated variance of the residuals (errors) from the model.

- **AIC (Akaike Information Criterion) = 14361.32**:
- **AICc (corrected AIC) = 14361.39**:
- **BIC (Bayesian Information Criterion) = 14385.45**:

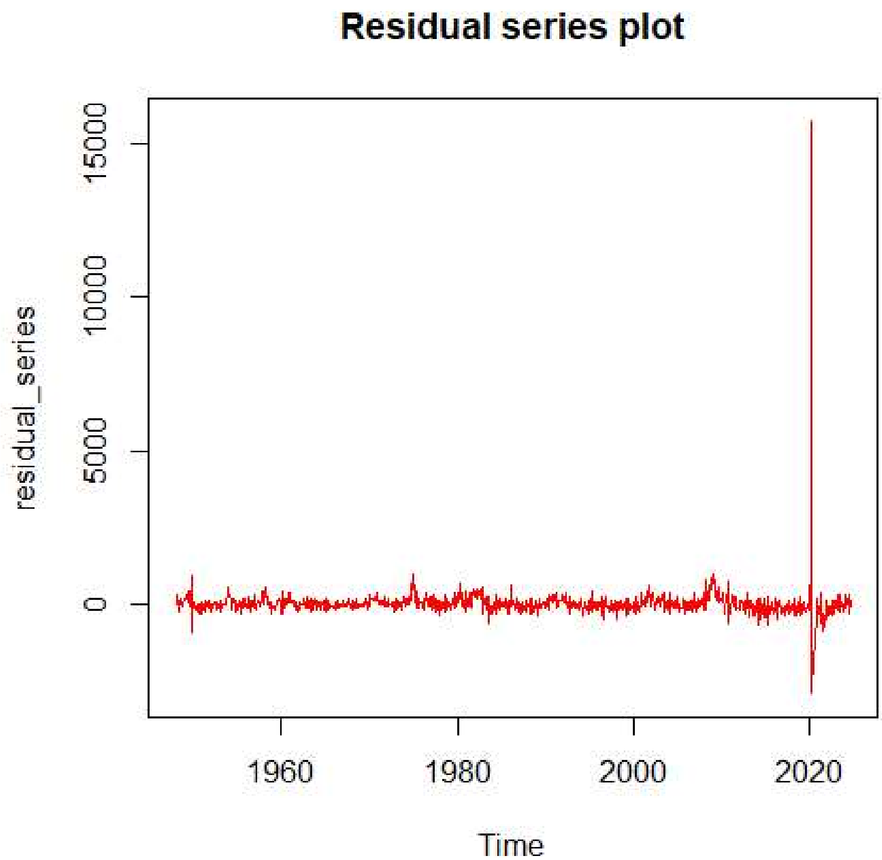
  These are used to compare different models; lower values generally indicate a better model fit. AICc is similar to AIC but includes a correction for small sample sizes.

11. Residual analysis

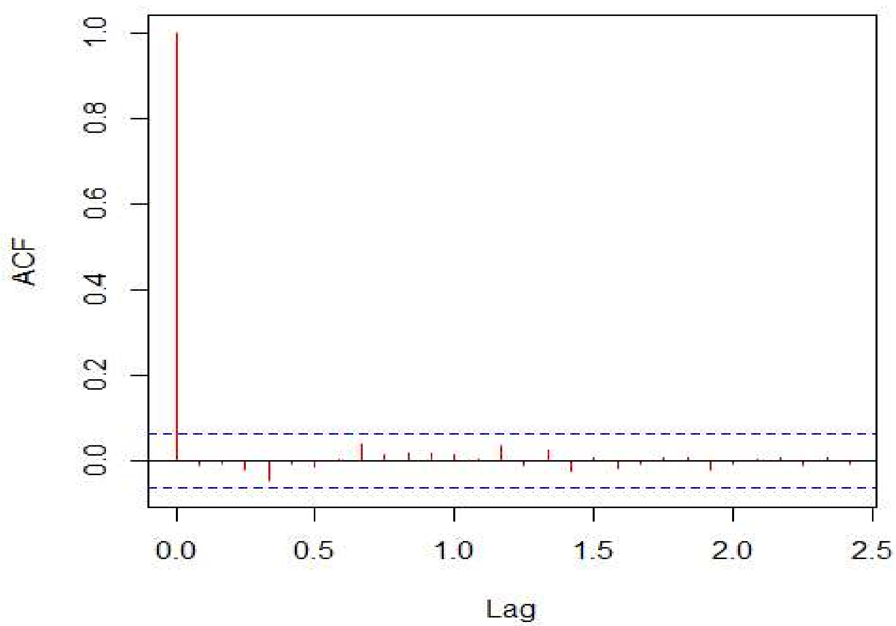
R-Code:

```
residual_series = residuals(model)
print(residual_series)
plot(residual_series,col="red",main="Residual series plot")

acf(residual_series,main="ACF Plot for residual series",col="red")

## qq plot
qqnorm(residual_series)
qqline(residual_series, col = "red") ## add a reference line ##
```

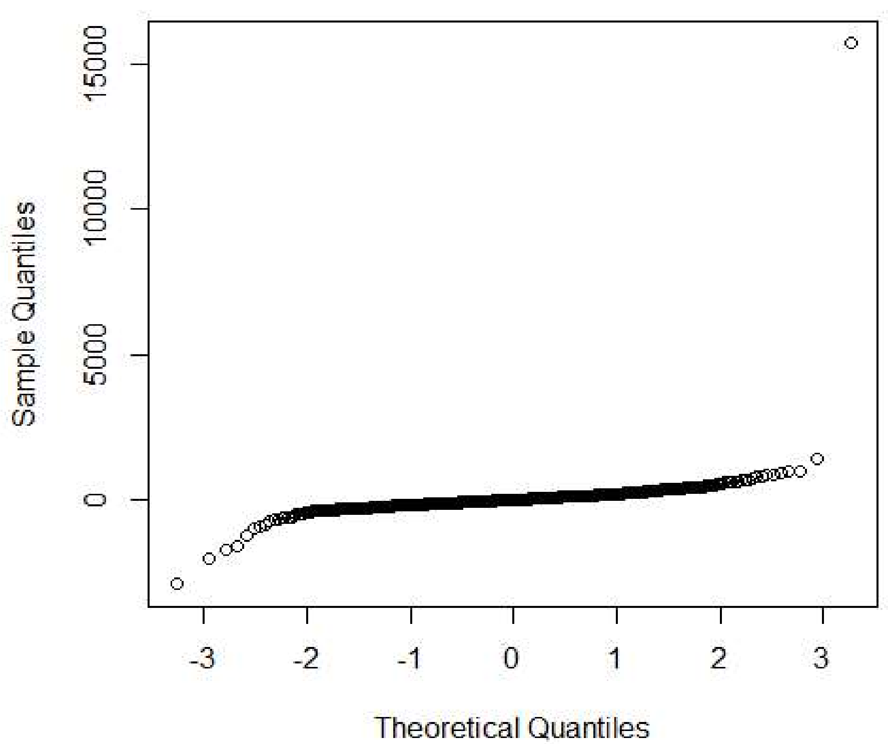
Output:

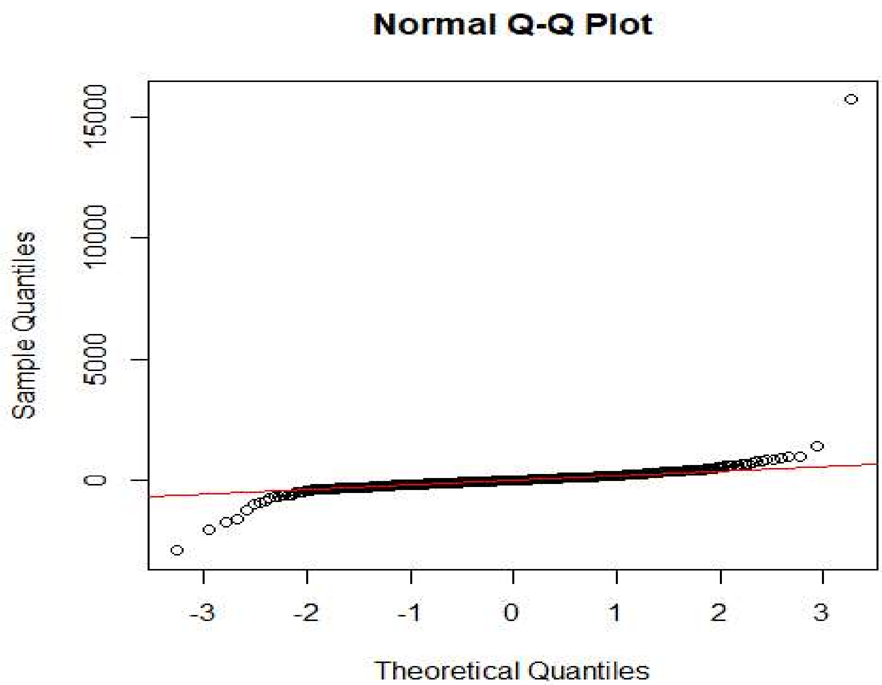

Residual series plot
residual_series
15000
10000
5000
0
1960
1980
2000
2020
Time

ACF Plot for residual series



Normal Q-Q Plot





Interpretation:

a. **Residual plot:** The plot shows that the residuals seem relatively stable and close to zero. This shows that the model's predictions were quite accurate during this period. Around 2020, there is significant spike in the residuals, which indicates that the model's predictions were off by a large margin because of Covid-19.

b. **ACF plot for residual series**: At lag 0, the autocorrelation is 1.0, meaning each residual is perfectly correlated with itself. For all other lags, the red bars are close to zero. The ACF plot shows that the residuals from the ARIMA (2,1,2) model do not have significant autocorrelation at any lag. This means that the residuals are essentially random noise, indicating that the model has successfully captured the patterns in the data.

c. **Normal QQ plot:** The Q-Q plot helps us check if the residuals from our ARIMA model are normally distributed. In this plot we see that for most of the middle values, the residuals seem to follow a normal distribution since the points lie close to the diagonal line.

d. **Updated Normal QQ plot (Add a reference line):** In the plot the red line represents the line where the points would lie if the residuals were perfectly normally distributed. For most of the middle values, the residuals seem to follow a normal distribution since the points lie close to the red reference line. This indicates that the model captures the main structure of the data but leaves some unusual points unexplained.

12. BOX plot

The Box-Ljung test is used to check for the presence of autocorrelation in the residuals of a time series model.

R-Code:

```
Box_test=Box.test(residual_series, lag = 10, type = "Ljung-Box");Box_test
if(Box_test$p.value<0.05)
{
cat("There is significant autocorrelation in the residuals.\n")
}else{
cat("autocorrelation is not present in the residuals.\n")
}
```

Output:

```
Box-Ljung test

data:  residual_series
X-squared = 4.4933, df = 10, p-value = 0.9224

autocorrelation is not present in the residuals.
```

Interpretation:

Here we see that the Box-Ljung test does not find significant autocorrelation in the residuals, it shows that the ARIMA (2,1,2) model is appropriately capturing the structure in the data. The residuals are behaving like white noise. The Box-Ljung test confirms that the residuals from the ARIMA model are free of autocorrelation.

13. Forecasting

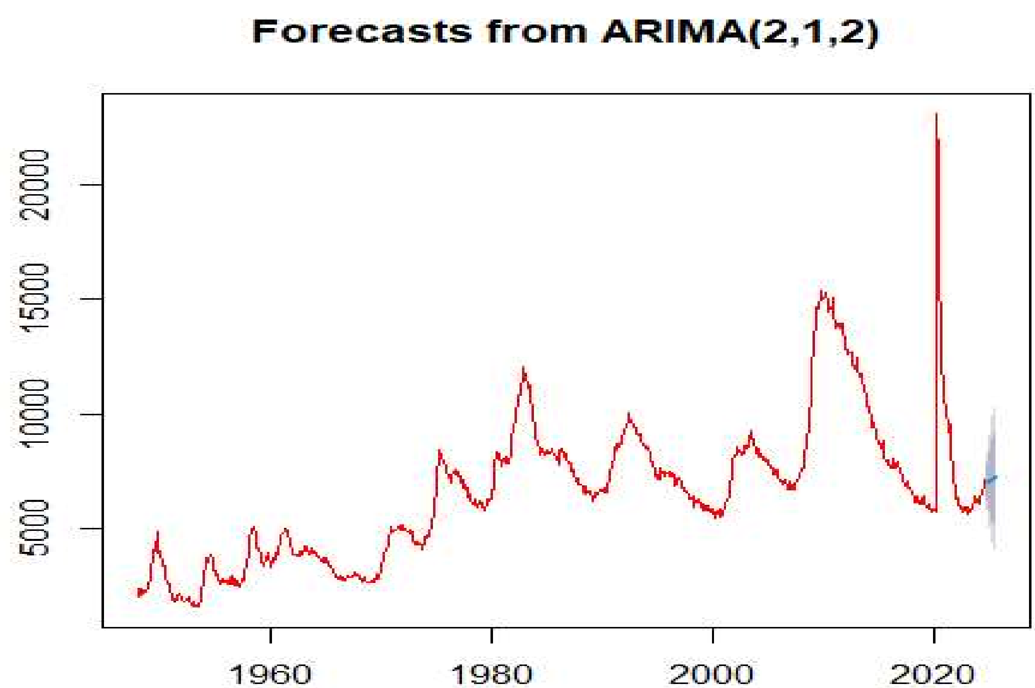
R-Code:

```
forecast_values = forecast(model, h = 10)
forecast_values
plot(forecast_values,col="red")
```

Output:

```
> forecast_values
         Point Forecast    Lo 80    Hi 80    Lo 95     Hi 95
Nov 2024       7044.248 6292.623 7795.874 5894.736  8193.760
Dec 2024       7063.125 5972.329 8153.920 5394.897  8731.353
Jan 2025       7107.645 5801.611 8413.680 5110.238  9105.053
Feb 2025       7131.673 5637.930 8625.416 4847.190  9416.156
```

Mar 2025	7167.203	5526.290	8808.116	4657.643	9676.763
Apr 2025	7192.268	5419.211	8965.326	4480.612	9903.925
May 2025	7222.161	5337.254	9107.068	4339.444	10104.878
Jun 2025	7246.437	5260.194	9232.680	4208.741	10284.133
Jul 2025	7272.419	5197.008	9347.829	4098.352	10446.485
Aug 2025	7295.155	5138.424	9451.886	3996.720	10593.590



Interpretation:

The point forecasts show the gradual increase in the unemployment level over the next ten months.

Lo 80 and Hi 80: The lower and upper bounds of the 80% prediction interval. There is an 80% probability that the actual unemployment level will fall within this range.

Lo 95 and Hi 95: The lower and upper bounds of the 95% prediction interval. There is a 95% probability that the actual unemployment level will fall within this range.

The 80% prediction intervals are narrower than the 95% prediction intervals, reflecting the different levels of confidence.